A Centralized Approach to Intrusion Detection System Management: Design, Implementation and Evaluation

Peter Pekarčík, Eva Chovancová, Martin Chovanec, Martin Štancel

Department of Computers and Informatics, Faculty of Electrical Engineering and Informatics, Technical University of Košice, Letná 9, 04200 Košice, Slovakia, peter.pekarcik@tuke.sk, eva.chovancova@tuke.sk, martin.chovanec@tuke.sk, martin.stancel@tuke.sk

Abstract: This paper presents the design, implementation, and evaluation of a novel control node for managing Intrusion Detection Systems (IDS). Existing IDS management solutions primarily focus on data visualization and lack comprehensive management capabilities, which are critical for effective intrusion detection. Our approach addresses these limitations by developing a centralized control node capable of managing multiple IDSs, providing real-time monitoring, configuration management, and enhanced security features. The control node uses SSH and SCP protocols for secure communication, supporting both centralized and distributed rule configurations. This flexible architecture enables efficient intrusion detection, even in high-traffic environments. The implemented system, featuring an intuitive graphical user interface (GUI) and robust management tools, supports both novice and advanced users, improving the overall usability and effectiveness of IDS management. Evaluation of the system under real-world conditions demonstrates that the control node reduces resource consumption, minimizes packet loss, and enhances detection efficiency by distributing workloads across multiple IDSs. The proposed solution offers a significant improvement in security management by enabling better control, monitoring, and configuration of IDSs, contributing to the overall security of the protected network.

Keywords: control node; intrusion; intrusion detection system; monitoring; security; tool

1 Intrusion Detection Systems

Intrusion detection systems (IDS) have become a critical area of focus due to the increasing frequency and sophistication of new attacks and intrusions. To understand it, it is necessary to deal with the definition of an intrusion. This can be interpreted as follows:

- according to [1], it is an activity aimed at gaining access to the system,
- according to [2], it is successfully gaining control of a computer system, compromising it or making it inaccessible,
- according to the papers [3] and [4], it is an attempt to violate file access rights, file integrity and computer system accessibility.

Intrusions occur primarily because of existing security vulnerabilities within the system. The biggest source of security vulnerabilities are vulnerabilities in programs and operating systems, which arise mainly when extending their functionality. Patel [5] and Paulins [6] categorize attacks into two types based on their origin:

- external attack,
- internal attack, i.e., the user is trying to gain or abuse access rights.

Several methods can be used to protect against such attacks, such as implementing firewalls, cryptography, intrusion detection systems, and attack prevention systems. According to papers [3], [7], [8] and [9], intrusion detection is defined as the process of monitoring events in an application, computer system, or network and analyzing them for potential intrusions. Paper [10] states that it is the most consistent technique in protecting against attacks. According to the definitions of intrusion detection systems in [6], [11], [12] and [13] it can be defined as security software or hardware that automates the process of detecting intrusions.

1.1 Classification of the IDS

Classification of intrusion detection systems based on the divisions given in the publications [1], [5], [14] and [15] is shown in Fig. 1.

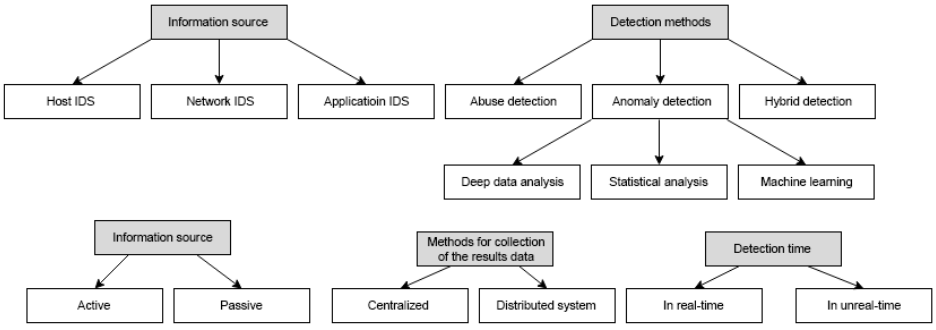

Figure 1

Classification of intrusion detection systems

Host IDS according to [16] and [17] monitors the characteristics and events of a single computer, e.g. packets sent to the computer, registers, log files, etc.

According to [10] and [18] network IDS monitors network traffic at the packet level, which it captures and then analyzes. The application IDS described in [5] consists in analyzing the logs of a specific application or analyzing its performance.

According to the authors [6], [11], [19], [20], the principle of how detection works is based on comparison with patterns of known attacks represented by rules. The main advantage of this detection method is its ability to reliably identify known attacks while maintaining a low false detection rate. The main disadvantage of this detection method, described in [13], [17] and [21], is the fact that new or modified types of attacks may not be detected due to the absence of a pattern or an unsatisfactory form of the pattern.

The anomaly detection described in [10], [13], [14] and [17] consists in defining a model of the standard behavior of the system. Based on the assumption that the attacker's behavior is different from that of a normal user, any deviation from the defined model is considered an attack.

According to [5], [11] and [14], data mining can enhance the efficiency of intrusion detection by reducing false positives through the identification of patterns, anomalies, changes, and significant events within the data. Techniques used in this type of data mining include e.g., cluster analysis. The disadvantages of this approach are high memory load and increased storage space requirement. The statistical analysis described in [5] and [14] consists of comparing two behavioral profiles – one that represents normal traffic and one that represents actual traffic.

Machine learning as defined in [5] and [14] is a method of analysis using artificial intelligence. The computer learns based on the analyzed data, thus increasing the detection efficiency. In [5], [24], [28] the authors defined the following terms as follows:

- IPS is an IDS with the ability to stop a potential attack, e.g., by changing the configuration of other security features, e.g., firewall.
- Distributed system is a system in which the resulting data is collected from multiple locations or sources.
- Centralized system is a system in which data are collected from a single location or source.
- Real-time detection is a detection method in which attacks are detected as long as the system, network or application is monitored by the IDS.
- When detecting in unreal time, the data is processed with a delay.

Currently, IDS are also implemented in mainstream security products. Attacks caught by IDS are mostly logged as events with varying priority. Events may or may not be attacks. As a general rule, the higher the priority of the event, the greater the likelihood of an attack.

We chose these IDSs because of the fact that we wanted to test multiple approaches. Suricata uses a multi-threaded architecture, which allows it to handle high-traffic environments more efficiently, whereas the single-threaded approach is used by Snort.

1.1.1 IDS Snort

Snort is a network IDS that uses a set of predefined rules for intrusion detection, i.e. it uses the principle of misuse detection for intrusion detection.

The Snort IDS is configured via the *"*.conf"* configuration file. The most important settings in terms of proper functionality are the configuration of event logging, the configuration of the path to the rules and the configuration of the rules themselves. More is found in [30].

1.1.2 IDS Suricata

Suricata is a network IDS that uses a set of predefined rules for intrusion detection, i.e., it uses the principle of misuse detection for intrusion detection [31].

1.1.3 Barnyard

Barnyard is a program that can process unified binary files in the *"unified2"* format. The biggest advantage of using this program is that IDS can concentrate all its resources on intrusion detection, leaving slow operations such as writing to the database to Barnyard [29].

1.2 IDS Management Tools

The Intrusion Detection Systems Management Tool is software designed to simplify working with IDS in any of the following ways [27]:

- simplification of IDS configuration,
- IDS station management,
- analysis of occurrence records,
- visualization of events,
- monitoring the occurrence of events.

There are currently several intrusion detection programs. The most well-known include Snorby, IDScenter and Log Siphon.

Snorby

According to [25], Snorby is a web application aimed at displaying statistical information about the occurrences of individual IDS events. The application supports various IDSs such as Snort, Suricata and Sagan. The most important functionalities of the application are the clear display of statistical data about

detected events presented through diagrams and tables, detailed information about detected events and the possibility to classify them, information and easy management of IDS stations and user management. In the main window of the application, the user can find the number of events for the selected period and a graph that visualizes:

- number of events captured by each sensor,
- number of events depending on severity,
- number of events depending on the protocol,
- pie chart with percentage display,
- percentage display of attack sources,
- percentage display of attack targets.

IDScenter

According to [26], IDS center is a program that simplifies the configuration and management of IDS Snort. The application offers a wide range of Snort configuration and management options. The solution offers the possibility to perform basic operations such as start or stop Snort, view detected events, test Snort configuration.

Log Siphon

Log Siphon is a program that can collect and analyze events in real-time. The program supports all systems that provide data via the Syslog standard and also IDS Snort and Suricata. The most important functionalities offered by the application include real-time monitoring of events, clear display of statistical data about the occurring events presented through charts and tables.

Evaluation of the Tools Analysis

Based on the analysis of the different tools, it is possible to create a table from which it can be deduced that there are currently few IDS management tools. The solutions are focused on visualization of collected data and not on monitoring and management. None of them provide the possibility of remote management of IDS stations to a sufficient extent, thus creating room for the emergence of a new solution. All the features are shown in Table 1.

Table 1
Comparison table of functional properties of the analysed tools

Feature	Snorby	IDScenter	Log Siphon
Fee	Yes	Yes	No
Easy to install	Yes	Yes	Yes
Intuitive GUI	Yes	Yes	No
Support of several IDSs	Yes	No	Yes

Support of several running IDSs	Yes	No	Yes
Platform independence	Yes	No	Yes
Monitoring of devices	Yes	Yes	Yes
Web interface	Yes	No	Yes
Remote control	No	No	No
Interactive configuration option	No	Yes	No
Automatic updates	Yes	Yes	No
E-mail notifications	Yes	Yes	Yes
Program logs	No	Yes	No
Direct configuration file edit	No	Yes	No
Statistics	Yes	No	Yes

2 Control Node for the Management of Intrusion Detection Systems

The design of an IDS management node can be divided into the following steps:

1) defining the requirements for the functionality of the future system,
2) defining the ways of using the system from the user's point of view,
3) design of the architecture and method of communication between the system components,
4) database design,
5) creation of the design of devices and their interconnections.

2.1 Basic Requirements for the Functionality of the System

Among the most important requirements for the future system is the management of IDS, which includes standard operations of adding, deleting and modifying IDS. The functionality is important because of the ability to manage and configure multiple IDSs. Another important requirement in this respect is a local database whose role is to store all data associated with IDS information.

IDS management is carried out remotely, so the need for secure communication over an insecure network must be addressed.

From the IDS management point of view, the most important requirement is to modify the IDS configuration. This includes the need for manual editing of the configuration file, aimed at more advanced users, and the need for interactive rule management, aimed at less advanced users. Rule management includes enabling, disabling, adding and deleting a rule.

The most important requirement in terms of IDS monitoring is to monitor the status and utilization of the IDS and to be alerted when an event occurs. Further requirements are: control node, authentication, GUI, IDS management, adding, deleting and modifying IDS, securing communication over an insecure network, modifying IDS configuration, modifying and changing the configuration file, rule management, enabling, disabling, adding and deleting a rule, remote IDS management (start, restart and shutdown), monitoring the status (on, off) and utilization of stations (CPU, RAM), event alerting, event occurrence report and local database.

2.2 Architecture of the Proposed Solution

The architecture of the proposed solution (Fig. 2) is diversified into three layers according to their focus:

- network traffic capture layer,
- intrusion detection layer,
- synchronization layer.

The role of the network traffic capture layer is to mirror the network traffic through the switch to each IDS. The goal of the intrusion detection layer is to detect intrusions using rules. The layer can consist of one or more IDSs. If multiple IDSs are used, each IDS may have:

- different rule configuration, allowing potentially more efficient detection by avoiding packet dropping due to IDS congestion or by using different rules for the same type of attack,
- the same configuration of rules, which guarantees a greater level of security in the event of an IDS failure due to a malfunction or attack, or in the event of a change in the configuration of the IDS, which requires a reboot of the IDS.

The combination of these options allows you to take advantage of both approaches, but rapidly increases the cost of the hardware.

The role of the synchronization layer, consisting of the control node and stations interconnected through the network, is the management and control of individual stations, i.e., the control and management of the intrusion detection layer. A description of the functionality of the control node is given in Section 2.1.

The goal of the proposed architecture is to improve the efficiency of intrusion detection, management and control of IDS stations and therefore ultimately and more security of the entire system.

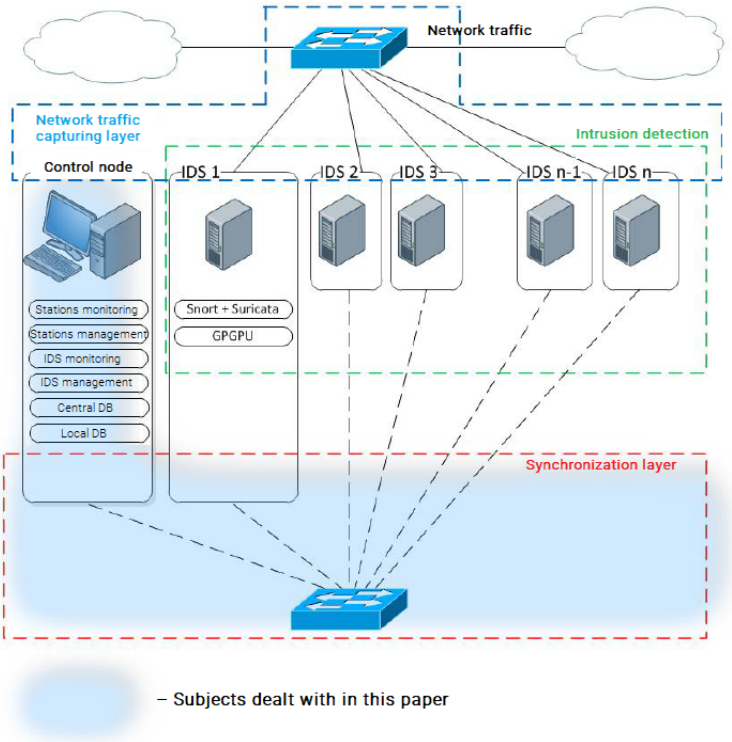

Figure 2
Proposed architecture

Two databases are placed on the control node:

- local database,
- central database.

The local database is used to store user and individual IDS data. It is described in more detail in subsection 2.3. The central database is a database designed to store all events from each IDS. The events are stored in a standardized form, which allows support for multiple IDS types. The standardized form is achieved by the use of Barnyard [29].

Detail of the proposed architecture is shown in Fig. 3.

The proposed architecture is based on a client-server architecture. The client (control node) sends requests to the server (IDS station), which processes them and sends them back to the client. The control node typically communicates with multiple stations.

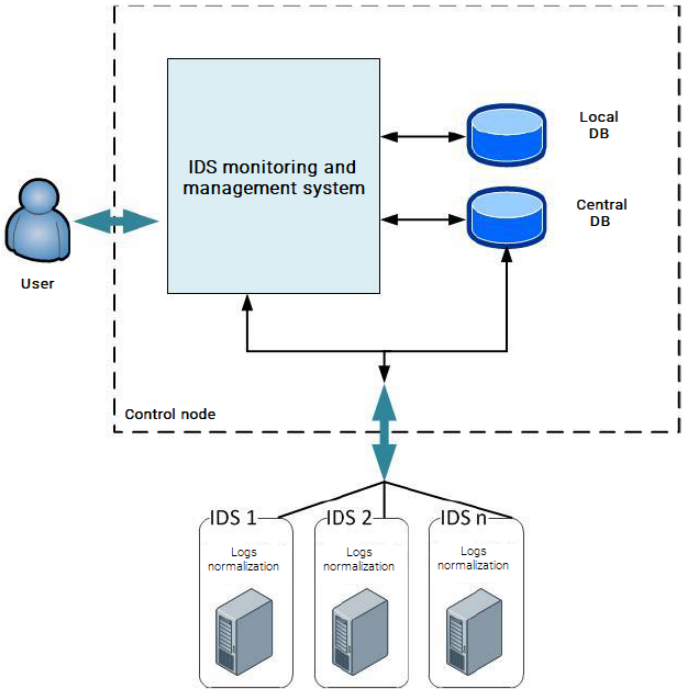

Figure 3
Detail of the proposed architecture

The SSH and SCP protocols can be preferably used for communication between the stations and the control node. The SSH protocol is used for monitoring the use of system resources of the station, monitoring the status of the station and IDS, remote management of the station and the IDS itself. The SCP protocol is used to send configuration files. The operations that need to be performed on the station side are implemented using scripts.

2.3 Local and Central Database Design

The local database is used to store login credentials, information about individual stations, and configuration files. During the implementation phase of the solution, new requirements from the target group arose and so the original design was modified.

The role of the central database is to store information about basic station characteristics and detected events. It consists of a structure that uses the Banyard program. It is complemented by the central table which stores the basic characteristics.

3 Control Node Implementation

The biggest challenge in implementation is solving the issue of concurrency, ensuring secure communication over an insecure network, as well as the overall security of the system and ensuring the ability to monitor and manage stations and IDS.

3.1 Network Communication

Communication over a computer network is a necessary element of the solution. Network communication is used for every action performed through the application. Therefore, it is necessary to choose the appropriate means, ensure proper functionality and ensure the security of the communication so that it cannot be misused.

The first type of communication is the IDS communication with the database, for which the MySQL Connector/JDBC library is used. The second type is the communication of the control node with the IDS, which is described in Section 3.1.1. The last type is the communication from the central database to the control node, described in Section 3.1.2.

3.1.1 Control Node Communication with IDS

The SSH and SCP protocols are used for communication between the control node and the IDS.

The SSH protocol is used for station monitoring (status, CPU usage, RAM usage, amount of data transferred, number of discarded packets), station management (shutdown, restart), IDS and Barnyard monitoring (status) and IDS and Barnyard management (configuration change).

The application of the SPC protocol is in sending files from the control node to the IDS and vice versa, which is used in IS and Barnyard management. With the SCP protocol, the sending and receiving of files between the control node and the station is implemented. This is particularly important because of the ability to edit IDS Suricata, IDS Snort and Barnyard files and upload them back.

3.1.2 Communication between the Central Database and the Control Node

The communication between the central database and the control node is implemented using the TCP/IP protocol. The data are retrieved and transmitted using a trigger.

3.2 System Security

The security of the system is implemented by means of authentication and security features. For successful user authentication, knowledge of correct login data – name and password, which must be entered through the application login window, is required. The login credentials are stored in the database. For increased security of sensitive data stored in the database system, login data is hashed and station data is encrypted.

3.3 GUI

The system includes a graphical user interface, which allows less computer-savvy users or users who are not familiar with the detailed management of ITS to work with the application. For this reason, the GUI is designed to be as simple and intuitive as possible.

The application contains only three program windows, namely the login window, the main application window (Fig. 4a) and a window that the user can open separately for each IDS, which displays basic information about the IDS (Fig. 4b).

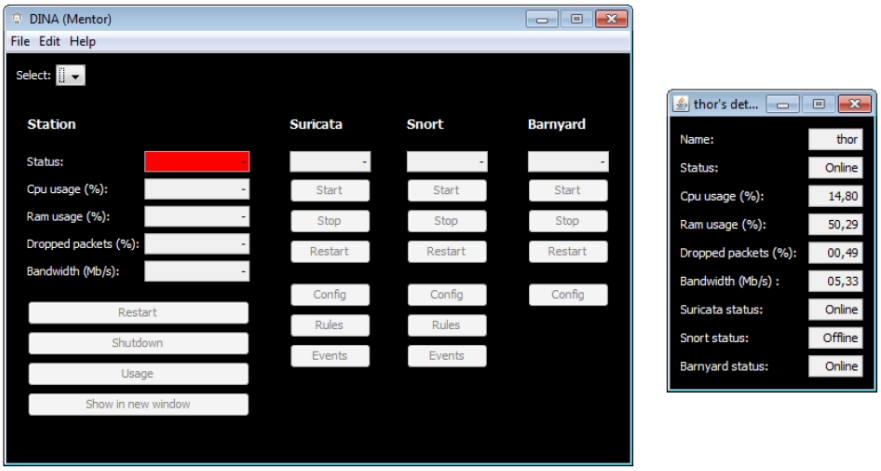

Figure 4
GUI, 5a (left), 5b (right)

4 Testing of the Proposed Solution

Testing of the application was carried out from the early stages of the implementation of the program to the final form. The aim was to detect bugs earlier and uncover any inconsistencies with the requirements of the target group.

4.1 Test Phase I – Major Deficiencies Identification

The objective of the Phase I test was to identify the major deficiencies in the system functionality and the ability to communicate with the station over the network. Another objective was to eliminate any GUI deficiencies so that the application could be handled by users who are less proficient or users who are not familiar with IDS management in detail.

During the test phase I, deficiencies in the form of faulty RAM count and faulty IDS operation start-up were detected. Two new requirements arose during Test Phase I, namely a requirement for basic Barnyard program management and a requirement for tracking the percentage of packets discarded.

4.2 Test Phase II – Communication Verification

The main objective of the test phase II was to verify the ability to communicate with multiple IDSs. Testing was, therefore, carried out on three devices, two of which were dedicated to the IDS and one to the control node. Another objective was to discover undetected bugs from the previous testing phase, or newly discovered bugs, and possibly implement new requirements.

During test phase II, the following deficiencies were identified: failure to detect basic characteristics and inability to apply changes to the IDS settings. At the same time, there were requirements for displaying basic data about individual IDS in separate windows, support for IDS Snort and support for multiple IDS Snort running on a single station.

4.3 Test Phase III – Functionality Verification

The main objective of the test phase III was to verify the overall functionality of the application and to eliminate any potential deficiencies. Testing phase III is purely a testing phase and thus no requests for functionality additions were accepted by the target group. Testing included artificially created situations that could lead to non-standard behavior of the application, or to its non-functionality or termination. During this testing phase, only one deficiency was identified in the form of the non-functionality of changing the IDS identifier.

5 Evaluation of the Proposed Architecture

Testing of the proposed architecture (see Section 2.2) was carried out on five devices (Fig. 5). The role of the communication generator and the attack generator is to simulate the network traffic on the VLAN200 network. Intrusion detection is implemented using two IDSs that sniff the communication from the VLAN200 network through two mirrored ports. The synchronization layer consists of a

control node and two IDSs control node and two IDSs that communicate with each other over VLAN100. The proposed system has been deployed on the control node.

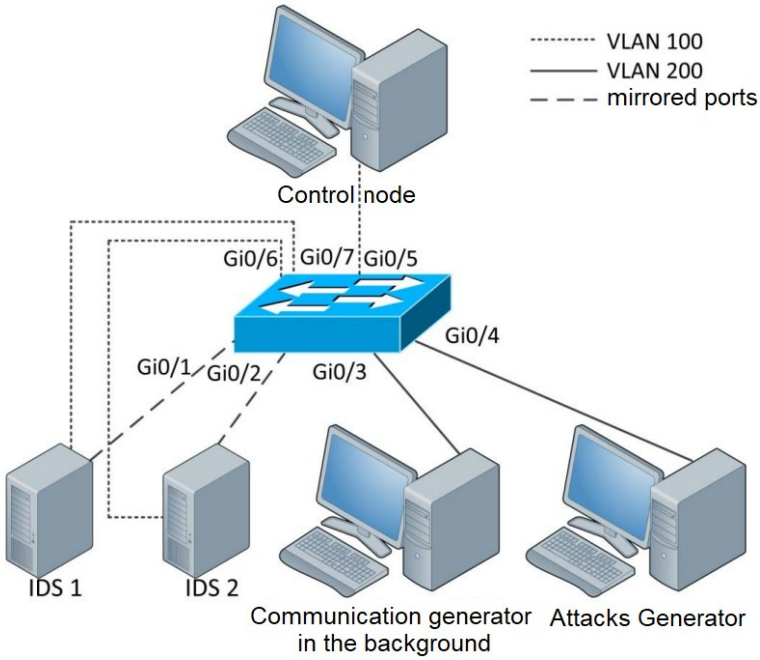

Figure 5
Connected devices used during the testing

The aim of the testing is to verify the effectiveness of the proposed architecture, i.e., to verify whether splitting the rules among multiple IDSs will reduce the IDS load and increase the detection efficiency, and whether using different types of IDSs or their configurations will make the intrusion detection more efficient.

The amount of data transferred in the simulated network traffic was at the level of 930 Mbps. The tests were carried out with both IDSs having the same configuration. First, one IDS was tested on which all rules were enabled. During the testing, the CPU and RAM utilization characteristics were measured and also the information about the number of discarded packets. After the test was completed, about half of the rules were disabled on this IDS and a second IDS was prepared with the rules that were disabled on the first IDS. On both IDSs, CPU and RAM utilization characteristics were measured during the test, as well as information on the number of discarded packets, as in the testing of one IDS.

The Suricata system was tested first. The CPU and RAM utilization of the Suricata IDS with all available rules enabled can be seen in Fig. 5 and the number of discarded packets is shown in Fig. 6.

The CPU and RAM utilization of the two Suricata IDSs with approximately equal distribution of available rules can be seen in Fig. 7 and the number of packets loss is shown in Fig. 8.

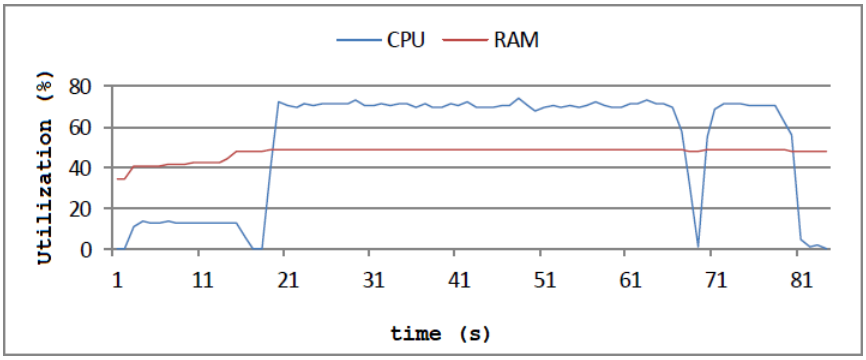

Figure 6
CPU and RAM utilization of IDS when using IDS Suricata

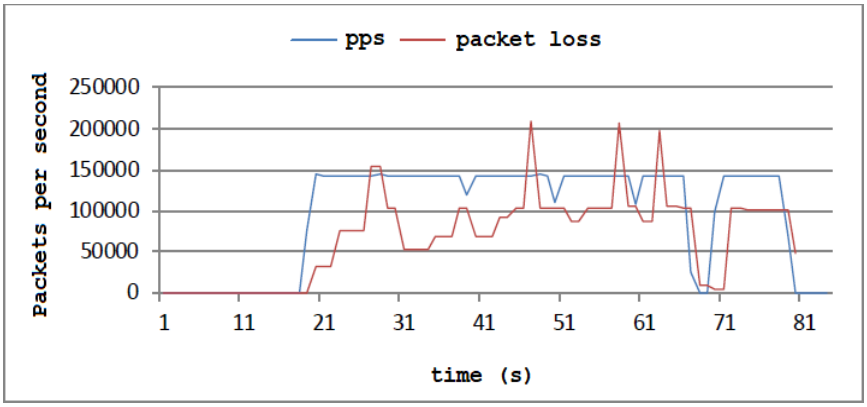

Figure 7
Total number of packets sent and packets discarded when using a single Suricata IDS

Based on the figures shown in Fig. 6, Fig. 7, Fig. 8 and Fig. 9, it can be concluded that in the case of splitting the rules between two IDSs, the load has also been distributed and the number of packets loss has been minimized, hence the overall security of the system has been increased.

IDS Snort was also tested in the same way. The CPU and RAM utilization of IDS Snort with all available rules enabled can be seen in Fig. 10 and the number of packets loss in Fig. 11.

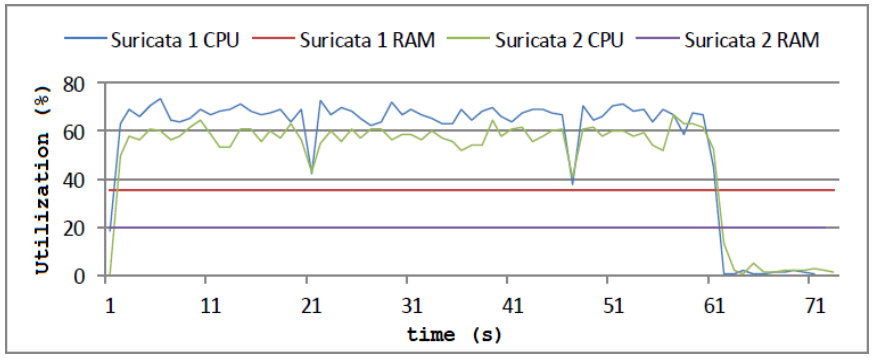

Figure 8
CPU and RAM utilization of the Suricata IDS using two IDSs with approximately uniform rule distribution

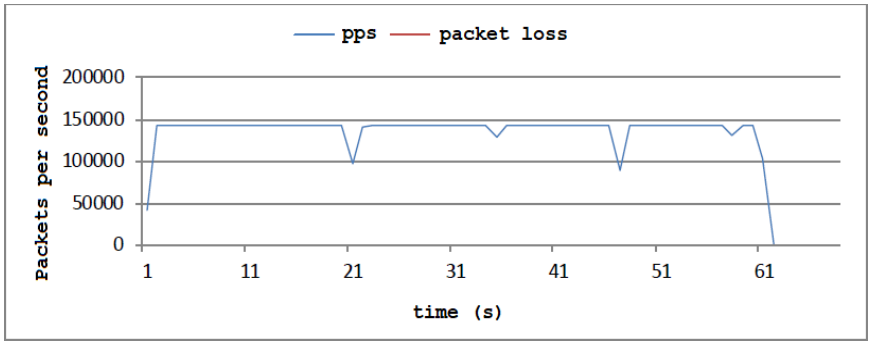

Figure 9
Total number of packets sent and packets loss when using two Suricata IDSs with an approximately even distribution of rules

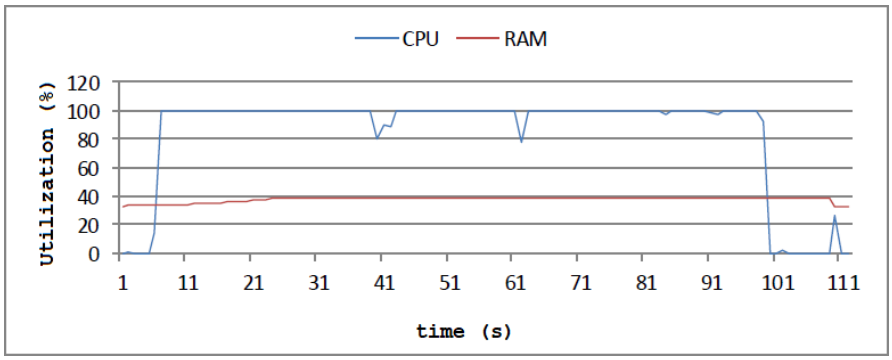

Figure 10
CPU and RAM utilization of IDS Snort with all rules activated

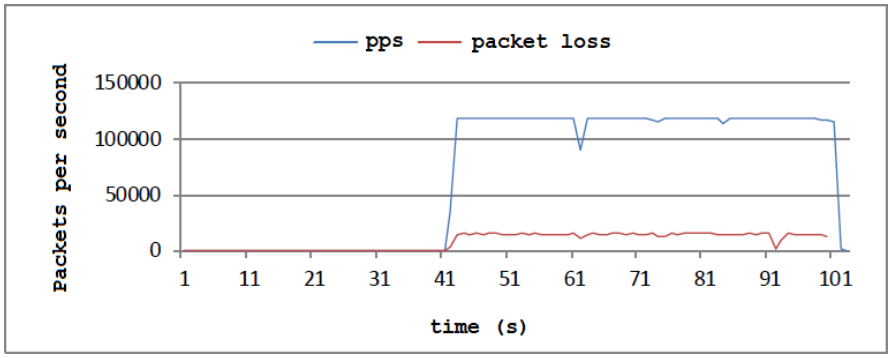

Figure 11
Total number of packets sent and packets loss when using a single Snort IDS

The CPU and RAM utilization of the two Snort IDSs with approximately equal distribution of available rules can be seen in Fig. 12 and the number of packets loss is shown in Fig. 13.

Based on the figures shown in Fig. 10, Fig. 11, Fig. 12 and Fig. 13, it can be concluded that in the case of splitting the rules between the two Snort IDSs, the load has also been distributed and the number of packets loss has been minimized, hence the overall security of the system has been increased.

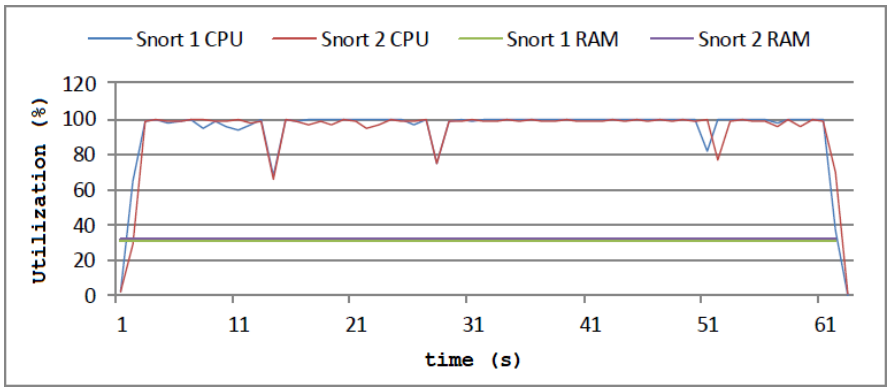

Figure 12
Total number of packets sent and discarded when using two Snort IDSs with approximately equal distribution of rules

The second test to validate the proposed architecture is a test in which attacks are generated while both IDSs are loaded. During testing, several attacks were encountered and detected by both IDSs. One example can be a DoS attack on the MSSQL database.

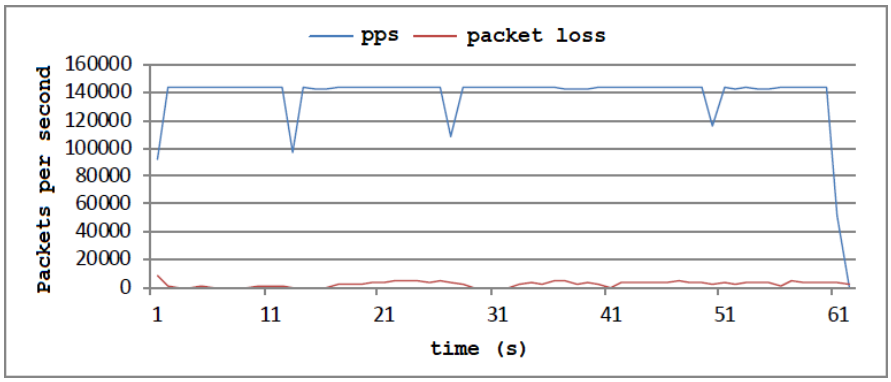

Figure 13
Total number of packets sent and packets loss when using two Snort IDSs with approximately equal distribution of rules

Furthermore, there were types of attacks that were detected only by the Suricata IDS, e.g., Full SYN Scan, a type of attack that detects which ports are open, and types of attacks that were detected only by the Snort IDS, e.g., an attack in which packets are sent that are subsequently scanned out of sequence. The reason for this phenomenon is the use of different rules by each IDS. The attack may not have been detected due to a non-conforming rule form or due to the absence of a rule. It can be concluded that the use of different IDS or different rules for the same type of attack increases the security of the system.

The solution provides the possibility of using the system for people who are not familiar with the issues. The solution also allows to fully perform all basic activities without the need to know the issues of these systems.

The solution performs continuous monitoring of devices and their basic characteristics (CPU utilization, RAM, amount of transferred data and number of discarded packets), which allows earlier detection of problems with the IDS connected station and thus a greater degree of security of the entire system.

The system supports the possibility of connecting multiple IDSs, each IDS can have a different configuration, which allows for faster detection and/or more efficient detection in case of using different rules for the same type of attack.

Conclusion

The analysis of tools for managing intrusion detection systems points to a major shortcoming of existing solutions, which is that the solutions focus on the visualization of collected data and not on monitoring and management of the IDS, which is critical in terms of system functionality. The analysis of these tools, their pros and cons are the basis for the design of the solution.

The result of the work is the implementation of a system for the management of intrusion detection systems, built on solid foundations, i.e., platform-independent modern object-oriented language Java in combination with a platform-independent relational database system with open-source MySQL. The solution is, in contrast to the currently existing solutions, oriented to monitoring and management of IDS. The system is designed and implemented in such a way that it does not require any non-standard tools for its operation, which greatly simplifies its deployment into operation and its initial configuration. The tool offers a simple and intuitive GUI, which makes it possible for people without in-depth knowledge of IDS to work with the system. On the other hand, it offers a wide range of functionality, making it suitable for more advanced users.

The main goal of the system is to increase the security of the system in addition to making the work of configuring and working with the IDS easier and faster. The control node contributes to the security of the system in several ways. The solution can work with several IDSs at the same time, supporting two types of IDSs - Snort and Suricata. Each IDS can have a different configuration, which allows to speed up the detection process. The use of different IDS systems or different rules for the same type of attack can lead to more efficient detection. Individual IDSs can also have the same rule configuration, which eliminates problems in the event of a failure of one of the IDSs. It is also possible to combine both approaches, which also means combining the benefits of these approaches. The system performs continuous monitoring of the devices and their basic characteristics (CPU utilization, RAM, amount of data transferred and number of discarded packets). This feature can help detect problems with the IDS station earlier. Unlike other similar solutions, the solution is able to monitor events in real-time and alert the user if an event occurs. The control node supports the Barnyard program, which is responsible for event logging, and thus the IDS can focus all of its resources on detecting anomalies. This feature of it contributes to more efficient detection of pranksters. The system offers the possibility of check node duplication, i.e., multiple check nodes can monitor the same IDS. All these features contribute significantly to the overall security of the system.

The implemented intrusion detection system management system allows logging records of CPU and RAM utilization, the percentage of discarded packets and the amount of data transferred to the database, so that this data can be used and processed by another program. In the future, the solution can be extended by adding a number of interactive IDS configurations, adding support for other IDSs and optimizing the system.

References

[1] N. Y. Khan, B. Raud, K. Ahmed: Comparative study of intrusion detection system and its Recovery mechanism. 2010 The 2nd International Conference on Computer and Automation Engineering (ICCAE), iss. 5, pp. 627-631, 2010

[2] M. Ali Aydin, A. H. Zaim, K. G. Ceylan: A hybrid intrusion detection system design for computer network security, in: Computers & Electrical Engineering, Vol. 35, iss. 3, pp. 517-526, 2009

[3] H. Liao et al.: Intrusion detection system: A comprehensive review, in: Journal of Network and Computer Applications. Vol. 36, iss. 1, pp. 16-24, 2013

[4] V. Marinova-Boncheva: A Short Survey of Intrusion Detection Systems, in: Problems of Engineering Cybernetics and Robotics, iss. 58, pp. 23-30, 2007

[5] A. Patel et al.: An intrusion detection and prevention system in cloud computing: A systematic review, in: Journal of Network and Computer Applications. Vol. 36, iss. 1, pp. 25-41, 2013

[6] N. Paulins: An agent-based hybrid intrusion detection system, in: Research for Rural Development - International Scientific Conference. Vol. 1 pp. 191-195, 2011

[7] L. Vokorokos, A. Baláž, B. Madoš: Application Security through Sandbox Virtualization, in Acta Polytechnica Hungarica, Vol. 12, iss. 1, pp. 83-101, 2015

[8] J. Juhár, L. Vokorokos: Separation of Concerns and Concern Granularity in Source Code, Proceedings of IEEE 13th International Scientific Conference on Informatics, Poprad, Slovakia, pp. 139-144, 2015

[9] S. A. N. Meisam, A. A. Mohammad: A distributed multiapproach intrusion detection system for web services, in: Proceedings of the 3rd international conference on Security of information and networks, pp. 238-244, 2010

[10] F. I. Shiri et al.: A parallel technique for improving the performance of signature-based network intrusion detection system, in: Communication Software and Networks (ICCSN), pp. 692-696, 2011

[11] P. Louvieris et al.: Effects-based feature identification for network intrusion detection, in: Neurocomputing. Vol. 121, pp. 265-273, 2013

[12] S. Shamshirband et al.: An appraisal and design of a multi-agent system based cooperative wireless intrusion detection computational intelligence technique, in: Engineering Applications of Artificial Intelligence. Vol. 26, iss. 9, pp. 2105-2127, 2013

[13] C. Amza et al.: Hybrid network Intrusion Detection, in: Intelligent Computer Communication and Processing (ICCP), pp. 503-510, 2011

[14] A. E. Nawal, S. F. Osama: Divided two-part adaptive intrusion detection system, in: Wireless Networks. Vol. 19, iss. 3, 2013

[15] M. Tomášek et al.: Intrusion detection system based on system behavior, in: Applied Machine Intelligence and Informatics (SAMI), pp. 271-275, 2012

[16] F. Sabahi, A. Movaghar: Intrusion Detection: A Survey, 3rd International Conference on Systems and Networks Communications, pp. 23-26, 2008

[17] F. A. Bin Hamid Ali, Yee Yong Len: Development of host based intrusion detection system for log files, in: Business, Engineering and Industrial Applications (ISBEIA), pp. 281-285, 2011

[18] Y. Lin et al.: The Design and Implementation of Host-Based Intrusion Detection System, in: Intelligent Information Technology and Security Informatics (IITSI), pp. 595-598, 2010

[19] S. A. Aneetha et al.: Hybrid network intrusion detection system using expert rule based approach, in: Proceedings of the Second International Conference on Computational Science, Engineering and Information Technology, pp. 47-51, 2012

[20] I. Corona et al.: Adversarial attacks against intrusion detection systems: Taxonomy, solutions and open issues, in: Information Sciences, Vol. 239, pp. 201-225, 2013

[21] Yu-Xin Ding et al.: Research and implementation on snort-based hybrid intrusion detection system, in: Machine Learning and Cybernetics, Vol. 3, pp. 1414-1418, 2009

[22] G. Kim et al.: A novel hybrid intrusion detection method integrating anomaly detection with misuse detection, in: Expert Systems with Applications, Vol. 41, iss. 4, pp. 1690-1700, 2014

[23] L. Vokorokos, A. Pekár, N. Adám, P. Darányi: Yet Another Attempt in User Authentication, in Acta Polytechnica Hungarica, Vol. 10, iss. 3, pp. 37-50, 2013

[24] L. Vokorokos, A. Pekár, P. Feciľak: IPFIX Mediation Framework of the SLAmeter Tool, Proceedings of 2013 11th IEEE International Conference on Emerging Elearning Technologies and Applications, Stará Lesná, Slovakia, pp. 311-314, 2013

[25] What and Why of Snorby [online] [quoted on 27.10.2023] Available on the internet: <https://github.com/Snorby/snorby/wiki/What-and-Why-of-Snorby>

[26] Snorby [online] [quoted on 27.10.2023] Available on the internet: <https://snorby.org/>

[27] L. Schwartz et al.: Technológia elektronického podpisu, in: Advances in Electrical and Electronic Engineering, Vol. 3, iss. 3, pp. 48-51, 2004

[28] L. Vokorokos, B. Madoš, N. Adám, A. Baláž: Innovative Operating Memory Architecture for Computers using the Data Driven Computation Model, in Acta Polytechnica Hungarica, Vol. 10, iss. 5, pp. 63-79, 2013

[29] J. Beale et al.: Snort 2.1 Intrusion Detection, Second Edition, Syngress Publishing, Inc., second edition, 2004

[30] M. Roesch, C. Green: SNORT Users Manual 2.9.4. 2013

[31] W. Park, S. Ahn: Performance Comparison and Detection Analysis in Snort and Suricata Environment. In: Wireless Personal Communications, Vol. 94, pp. 241-252, 2017